ECE 417

Fall 2018

Machine Problem #2
Hidden Markov Model

Yichi Zhang, Qihao Wang, Qianwei Li

1. Introduction

In this machine problem, we will develop a speech recognizer by using the method of Hidden Markov Model. We divide this experiment by two part. The first part is “people dependent” part and the second part is “people independent” part. Then we update the parameter and our model and iterate for 10 times. After we training our model, we will use the test data and our own voice to calculate the accuracy of our model.

2. Algorithm

Extract and split data:

In this part, we use dictation to separate the input feature into people independent set and people dependent set and write all the function in our defined class. The first part is “people dependent” part and the second part is “people independent” part. In “people independent” part, we use all 75 data from three different people DG, LX as our training data, and LS and use 25 data from MH as our test data. In “people dependent” part, we collect 4 samples of each word, from each of the four training speakers. In this case, we use 80 data as our training data and the remaining 20 data as our test data. After we training our model, we will use the test data to calculate the accuracy of our model. Finally, we record our own voice as the test data to test our model in people independent part.

Training data :

- Initialization:

An HMM is normally identified with parameter set {π, A, B } where π is the initial state distribution, A is the transition probability matrix, and B is an observation matrix, which contains the emission probability. We use Gaussian as our likelihood function the equation to calculate the emission probability:

$$b_i(\vec{x_t}) = p(\vec{x_t}|q_t = i) = \frac{1}{(2\pi)^{D/2}|\Sigma_i|^{1/2}} e^{-\frac{1}{2}(\vec{x_t}-\vec{\mu_i})^T \Sigma_i^{-1} (\vec{x_t}-\vec{\mu_i})}$$

Because the likelihood function depends on μ and σ, the training problem becomes determining the parameters of { π, A, μ, Σ}.

In this part, we first initialize our Transition Matrix as

```
[[0.8 0.2 0.  0.  0. ]
 [0.  0.8 0.2 0.  0. ]
 [0.  0.  0.8 0.2 0. ]
 [0.  0.  0.  0.8 0.2]
 [0.  0.  0.  0.  1. ]]
```

And initialize the initial state distribution as π = [0.2, 0.2, 0.2, 0.2, 0.2].

μ can be initialized as the mean across the audio features for that word, and σ can be initialized as the covariance matrix across the audio features for that word.

- Forward-backward algorithm

We use the forward-backward algorithm to calculate the parameters α and β:

$$\alpha_t(i) = \Pr\left\{q_t = i, \vec{x_0}, \ldots, \vec{x_t}\right\}$$

$$\beta_t(i) = \Pr\left\{\vec{x_{t+1}}, \ldots, \vec{x_{T-1}} | q_t = i\right\}$$

- Forward Algorithm

In forward part, because α is small, in order to avoid overflow, we calculated the scaled version of α by the following equation:

$$\tilde{\alpha_t}(i) = \frac{\alpha_t(i)}{\prod_{\tau=1}^{t} g_\tau}$$

$$\bar{\alpha}_t(i) = b_i(\vec{x_t}) \sum_{j=0}^{N-1} \tilde{\alpha_{t-1}}(j) a_{ji}$$

$$g_t = \sum_{i=0}^{N-1} \bar{\alpha}_t(i)$$

$$\tilde{\alpha_t}(i) = \frac{1}{g_t} \bar{\alpha}_t(i)$$

Where $\alpha_0(i) = \Pr\left\{q_0 = i, \vec{x_0}\right\} = \pi_i b_i(\vec{x_0})$ and we pass the value of g as the parameter to the Backward function

- Backward Algorithm

In this part, we use the g which is passed from forward algorithm to calculate the scaled β to avoid the overflow by using the following equation:

$$\beta_t(i) = \Pr\left\{\vec{x_{t+1}}, \dots, \vec{x_{T-1}} | q_t = i\right\} \sum_{j=0}^{N-1} a_{ij} b_j(\vec{x_{t+1}}) \beta_{t+1}(j)$$

$$\tilde{\beta_t}(i) = \frac{\beta_t(i)}{\prod_{\tau=t+1}^{T} g_\tau}$$

where $\beta_{T-1}(i) = 1$

- Update parameter

In this part, we use the α and β calculated from forward-backward algorithm to update our parameter set {A, μ, Σ} and π is unchanged. In people dependent part, we have 16 files per word, and for people independent part, we have 15 files per word. For each model, we have to find 15 or 16 parameter sets { π, A, μ, Σ}

At first we use α and β to calculate the parameter γ and ξ by using the following equation

$$\gamma_t(i) = \Pr\left\{q_t = i | \vec{x_0}, \dots, \vec{x_{T-1}}\right\} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=0}^{N-1} \alpha_t(j)\beta_t(j)}$$

$$\xi_t(i,j) = \Pr\left\{q_t = i, q_{t+1} = j | \vec{x_0}, \dots, \vec{x_{T-1}}\right\} = \frac{\alpha_t(i) a_{ij} b_j(\vec{x_{t+1}}) \beta_{t+1}(j)}{\sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \alpha_t(i) a_{ij} b_j(\vec{x_{t+1}}) \beta_{t+1}(j)}$$

After we calculate the γ and ξ we can use them to update our parameter {A, μ, Σ}

$$a_{ij} = \frac{\sum_{\ell} \sum_{t} \xi_{t\ell}(i,j)}{\sum_{\ell} \sum_{t} \gamma_{t\ell}(i)}$$

$$\vec{\mu_i} = \frac{\sum_{\ell} \sum_{t} \gamma_{t\ell}(i) \vec{x_t}}{\sum_{\ell} \sum_{t} \gamma_{t\ell}(i)}$$

$$\sigma_{di}^2 = \frac{\sum_{\ell} \sum_{t} \gamma_{t\ell}(i)(x_{dt} - \mu_{di})^2}{\sum_{\ell} \sum_{t} \gamma_{t\ell}(i)}$$

And ℓ means the ℓ^{th} file.

- Iteration

and for each model, we update it 10 times. as we update more times, the accuracy increase and then converge.

- Test data

$$\hat{y} = argmax_y P_{Y|X}(y|X)$$

$$= argmax_y \frac{P(X,y)}{P(X)}$$

Because X is independent with y, we can factor out the P(x) and

$$\hat{y} = argmax_y P(X,y) = argmax_y P_{X|y}(X|y)$$

So we can use the forward algorithm to calculate the $\hat{y} = \sum_{j=0}^{N-1} \alpha_T(j)$ is the probability of the data sequence, X given the model parameters { π, A, μ, Σ}. Therefore we find

which label has the highest probability and then declare that label by compare $\sum_{j=0}^{N-1} \alpha_T(j)$. We can calculate $\sum_{j=0}^{N-1} \alpha_T(j)$ by the equation:

$$p(X|\Lambda) = \sum_{j=0}^{N-1} \alpha_T(j) = \prod_{t=1}^{T} g_t$$

$$\ln p(X|\Lambda) = \sum_{t=1}^{T} \ln g_t$$

Evaluate the model:

By using the equation the equation above, we compute the likelihood of a word utterance in test set given the model parameters for both part. We can calculate the accuracy the correct label by the total test data.

- people independent part:

For people independent part, we generate the output as the following:

```
[[0, 0, 0, 0, 0], [0, 1, 1, 1, 1], [3, 3, 3, 3, 3], [3, 3, 3, 3, 3], [4, 4, 4, 4, 0]]
[[1.  0.  0.  0.  0. ]
 [0.2 0.8 0.  0.  0. ]
 [0.  0.  0.  1.  0. ]
 [0.  0.  0.  1.  0. ]
 [0.2 0.  0.  0.  0.8]]
the accuracy of the speaker independent experiment is 72.0 %
```

That means all of word 0 (‘asr’) declared word 0(‘asr’), one of the word 1(‘cnn’) declares word 0(‘asr’), 5 of world 2 (‘dnn’) declare world 3(‘hmm’), and all of the word 3(‘hmm’) and four(‘tts’) have the correct label and our accuracy is 18/25 = 72%

- people dependent part:

For people dependent part, we generate the output as the following:

```
[[0, 0, 0, 0], [1, 1, 1, 1], [2, 2, 2, 2], [3, 3, 3, 3], [4, 4, 4, 4]]
[[1. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0.]
 [0. 0. 1. 0. 0.]
 [0. 0. 0. 1. 0.]
 [0. 0. 0. 0. 1.]]
the accuracy of the speaker dependent experiment is 100.0 %
```

That means all of the data have correct label.

- Own recorded part:

We record our own voice as the test data and generate the output for people independent model part as the following output:

```
[[0, 0, 0, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0], [3, 3, 3, 3, 3], [4, 4, 4, 4, 4]]
[[1. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0.]
 [0. 0. 0. 0. 1.]]
the accuracy of the user input experiment is 60.0 %
```

That means all of the word 0 (‘asr’), word 3 (‘hmm’) and word 4 (‘tts’) have the correct label, but word 1(‘cnn’) and word 2 (‘dnn’) declared word 0 (‘asr’)

Result Analysis :

Our people dependent part have a very high accuracy, we believe that is because people speak one word 5 times in similarly way so we can easily distinguish them. But for people independent part, each people speak one word differently, so it might difficult to distinguish them. Also, some words like ‘cnn’, ‘dnn’ and ‘hmm’ are similar to each other so it might difficult to distinguish them, but words like ‘asr’ and ‘tts’ are different from other words so we can distinguish them well. In order to increase the accuracy, we can use the other feature instead of mfcc or using a different likelihood function to update the parameter and we discussed how to do it in the ‘Extra Credit’ part.

Extra Credit:

- New model

On the top of the experiment, we have a Gaussian Mixture Model added to the original HMM model. In the normal HMM, we use the Gaussian distribution to calculate the observation probability. That is $b_j(x) = P(x \mid q_t = j)$. In the Gaussian Mixture Model, an M-component model is an approximate density function:

$$b_j(\mathbf{x}) = p(\mathbf{x} \mid S = j) = \sum_{m=1}^{M} c_{jm} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_{jm}, \boldsymbol{\Sigma}_{jm})$$

Notice that in this model, we have “M” components of Gaussian distributions and the final probability is the sum of M components. For each component, the mean and covariance matrix are given as:

$$\vec{\mu}_{jm} = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T} \vec{x}_{tl} \gamma_{tl}(j,m)}{\sum_{l=1}^{L} \sum_{t=1}^{T} \gamma_{tl}(j,m)}$$

$$\Sigma_{jm} = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T} (\vec{x}_{tl} - \vec{\mu}_{jm})(\vec{x}_{tl} - \vec{\mu}_{jm})^T \gamma_{tl}(j,m)}{\sum_{l=1}^{L} \sum_{t=1}^{T} \gamma_{tl}(j,m)}$$

Also, in each iteration, we also need to update c value used in the calculation of density function.

$$c_{jm} = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T} \gamma_{tl}(j,m)}{\sum_{m'=1}^{M} \sum_{l=1}^{L} \sum_{t=1}^{T} \gamma_{tl}(j,m')}$$

Therefore in summary, we have a new version of the observation likelihood matrix that will be used in forward/backward algorithm. After we have calculated all α values and β values, we will generate γ accordingly and therefore calculate the updated mean and covariance. By iterations through training data, we get the model and when we test our GMM-Speaker Independent model on the test data set, the accuracy is given as following figure: The accuracy is 76%, which is 4% higher than the HMM model in the previous part.

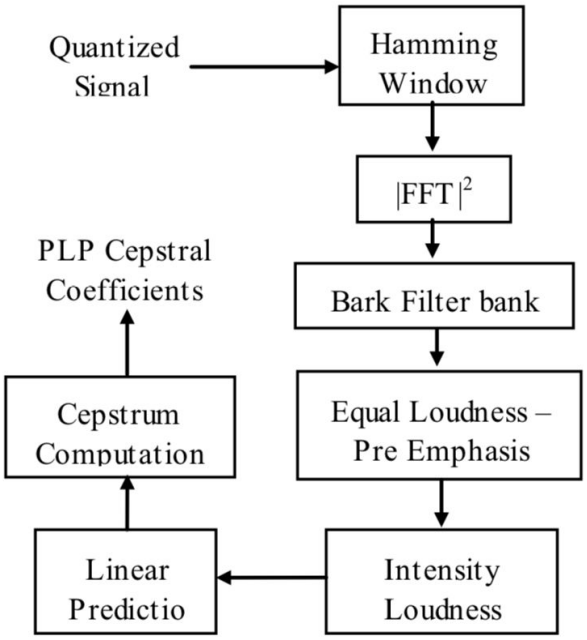
```
[[0, 0, 0, 0, 0], [0, 1, 1, 1, 1], [3, 3, 3, 3, 3], [3, 3, 3, 3, 3], [4, 4, 4, 4, 4]]
[[1.  0.  0.  0.  0. ]
 [0.2 0.8 0.  0.  0. ]
 [0.  0.  0.  1.  0. ]
 [0.  0.  0.  1.  0. ]
 [0.  0.  0.  0.  1. ]]
the accuracy of the speaker independent experiment is 76.0 %
```

- New Feature:

We can use PLP Cepstral Coefficients instead of MFCC for our model, The first step is a conversion from frequency to bark, which is a better representation of the human hearing resolution in frequency. The bark frequency corresponding to an audio frequency is :

$$\Omega(\omega) = 6ln\left[\frac{\omega}{1200\pi} + \left[\left(\frac{\omega}{1200\pi}\right)^2 + 1\right]^{0.5}\right]$$

The following figure shows detail steps of PLP computation:



Reference:

[1] Hiroshi Shimodaira and Steve Renals

https://www.inf.ed.ac.uk/teaching/courses/asr/2016-17/asr03-hmmgmm-handout.pdf

[2] Namrata Dave : Feature Extraction Methods LPC, PLP and MFCC In Speech Recognition

https://pdfs.semanticscholar.org/0b44/265790c6008622c0c3de2aa1aea3ca2e7762.pdf